



Introduction to Python

- **Getting Started with Python**
- **Model -> Python**
- **Creating your first Python script**
- **Running a script in ArcMap**

What is Python?

“Python is an easy to learn, powerful language... (with) high-level data structures and a simple but effective approach to object-oriented programming. Python’s elegant syntax and dynamic typing...make it an ideal language for scripting...in many areas and on most platforms.” –python.org



Scripting language of ArcGIS

**Free, cross-platform, easy to learn,
widely useful, great community**

Why use Python and ArcGIS?

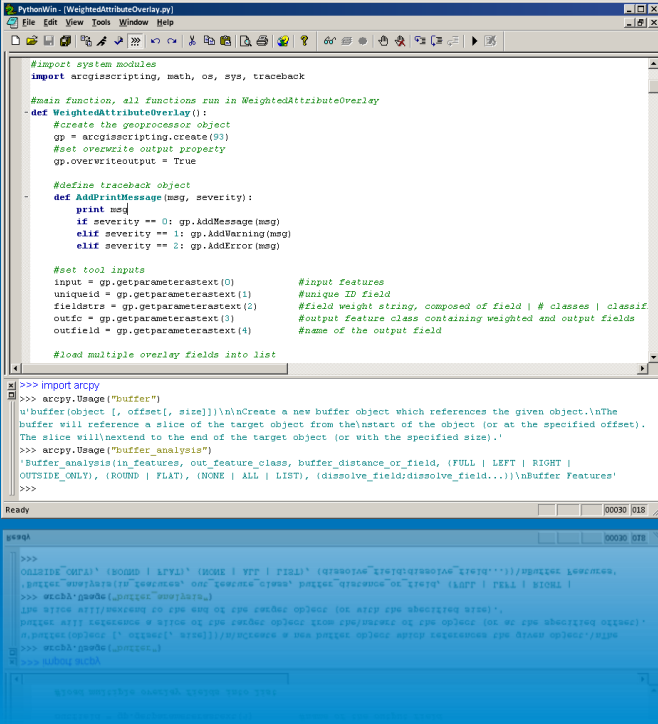
Automate repetitive tasks

Develop custom tools

Add geoprocessing to web applications

Customize Desktop apps

Extend the capabilities of ArcGIS



```
#import system modules
import arcpy, math, os, sys, traceback

#Main function, all functions run in WeightedAttributeOverlay
def WeightedAttributeOverlay():
    #create the geoprocessor object
    gp = arcpyscripting.create(93)
    #set overwrite output property
    gp.overwriteoutput = True

    #define traceback object
    def AddPrintMessage(msg, severity):
        print msg
        if severity == 0: gp.AddMessage(msg)
        elif severity == 1: gp.AddWarning(msg)
        elif severity == 2: gp.AddError(msg)

    #set tool inputs
    input = gp.getparameterastext(0) #input features
    uniqueid = gp.getparameterastext(1) #unique ID field
    fieldstrs = gp.getparameterastext(2) #field weight string, composed of field | # classes | classif
    outfc = gp.getparameterastext(3) #output feature class containing weighted and output fields
    outfield = gp.getparameterastext(4) #name of the output field

    #load multiple overlay fields into list

>>> import arcpy
>>> arcpy.Usage("buffer")
u'buffer(object [, offset[, size]])\n\nCreate a new buffer object which references the given object.\n\nThe buffer will reference a slice of the target object from the\start of the object (or at the specified offset).\n\nThe slice will\move to the end of the target object (or with the specified size).
>>> arcpy.Usage("buffer_analysis")
'buffer_analysis(in_features, out_feature_class, buffer_distance_or_field, (FULL | LEFT | RIGHT |
OUTSIDE_ONLY), (ROUND | FLAT), (NONE | ALL | LIST), (dissolve_field:dissolve_field...))\n\nBuffer Features'
>>>
```

Python 101

Where do I write Python code?

Python file is text with .py extension

IDE like **PyScripter**, **Wing IDE (\$)**, **PythonWin**

Python window in ArcGIS

How do I run? ...Double-click, IDE, ArcGIS

What are variables?

A name that stores a value; assigned using =

```
input = "C:/Data/Roads.shp"
distance = 50
both = [input, distance]

# Variables act as substitutes for raw values
arcpy.Buffer_analysis(input, "Roads_buffer.shp", distance)
```

Python 101

Python has logic for testing conditions

if, else statement

Colon at end of each condition

Indentation determines what is executed

`==` tests equality; other operators like `>`, `<`, `!=`

```
var = "a"
if var == "a":
    # Execute indented lines
    print("variable is a")
else:
    print("variable is not a")
```

Python 101

Techniques for iterating or **looping**

While loops, for loops

Colon at end of statement

Indentation determines what is executed

```
x = 1
while x < 5:
    print(x)
    x = x + 1

x = [1, 2, 3, 4]
for num in x:
    print(num)
```

Python building blocks

Function: a defined piece of functionality that performs a specific task; requires arguments

Module: a Python file where functions live; `import`

Package: a collection of related modules

`math.sqrt(100) ... 10`

Python Standard Library / Built-ins

`os, sys, math, datetime, urllib2`

ArcPy

Site package that adds ArcGIS functionality to Python

Access to 800+ geoprocessing tools

Functions, classes and modules

- Helper functions like **ListFeatureClasses**, **Describe**

- Classes that can be used to create complex objects like **SpatialReference**, **FieldMap**

- Modules that provide specialized functionality like **Mapping**, **SpatialAnalyst**, **NetworkAnalyst**, **DataAccess**

Enhancement of *arcpy* module (pre-10.0)

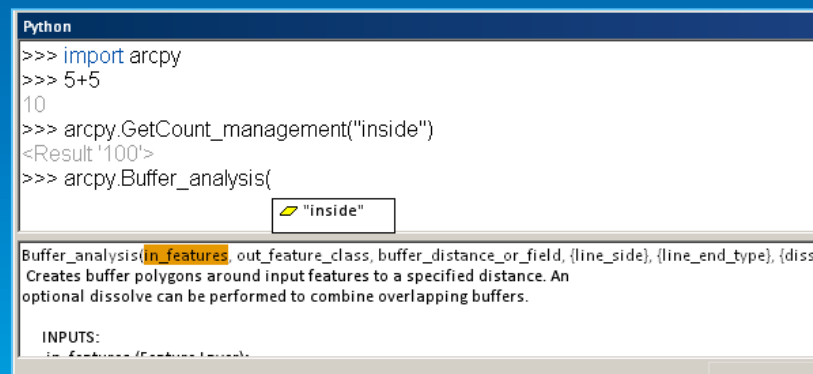
- Your old scripts **will** work

ArcGIS Python window

Embedded, interactive Python command line

Access to Python and modules within
ArcGIS applications

Great for experimenting with Python code
and learning tool syntax

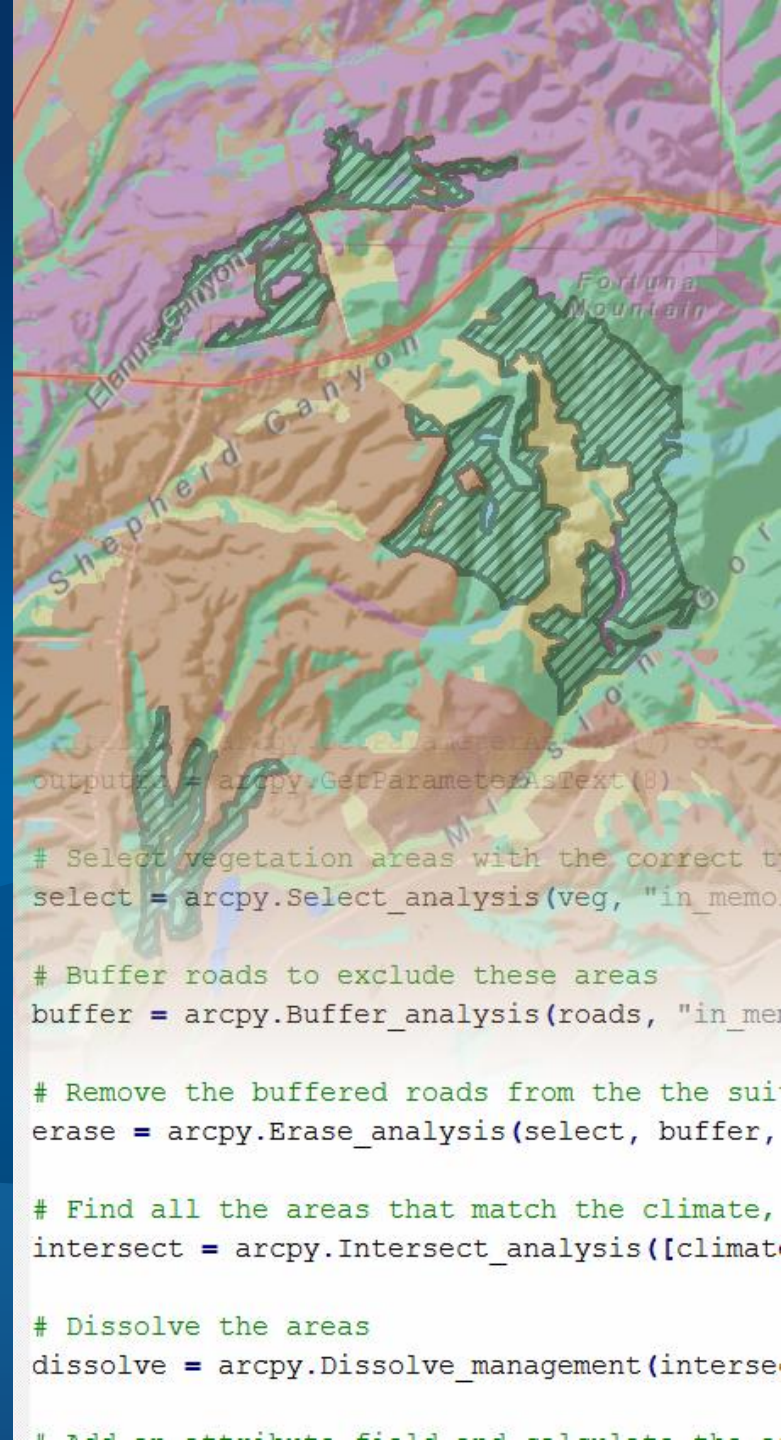


```
Python
>>> import arcpy
>>> 5+5
10
>>> arcpy.GetCount_management("inside")
<Result '100'>
>>> arcpy.Buffer_analysis(
    "inside"
```

Buffer_analysis(**in_features**, out_feature_class, buffer_distance_or_field, {line_side}, {line_end_type}, {dissolve})
Creates buffer polygons around input features to a specified distance. An optional dissolve can be performed to combine overlapping buffers.

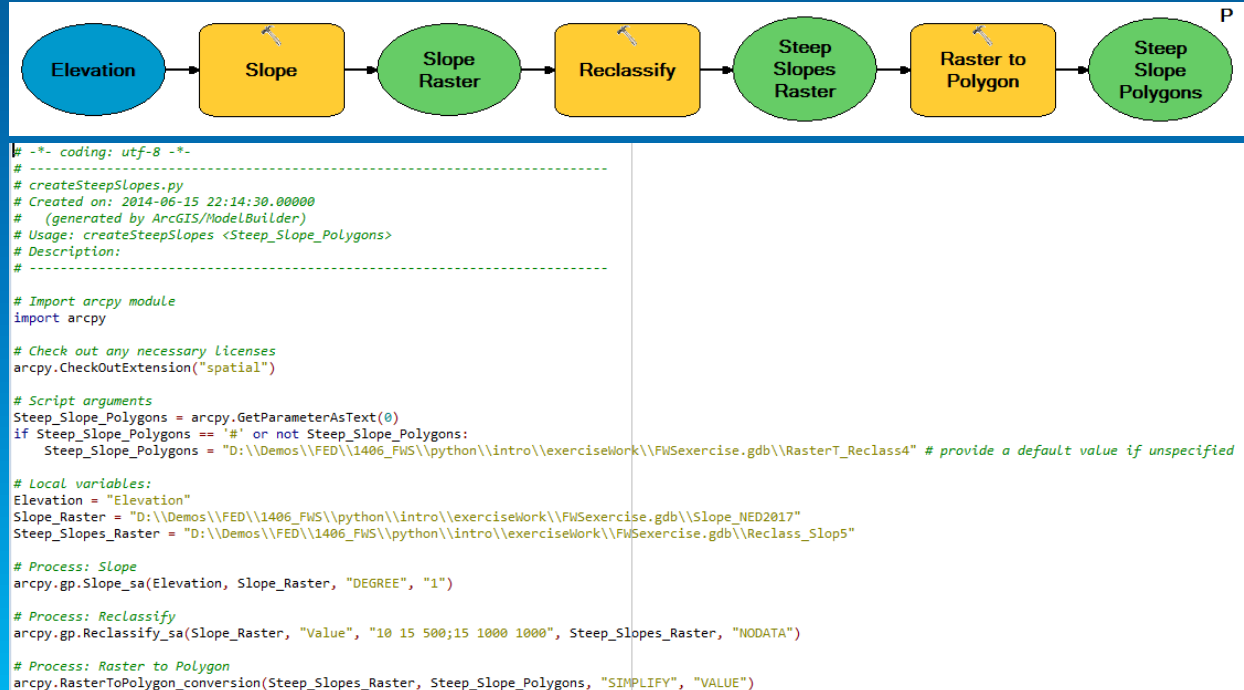
INPUTS:
in_features (Feature Layer)

Introduction

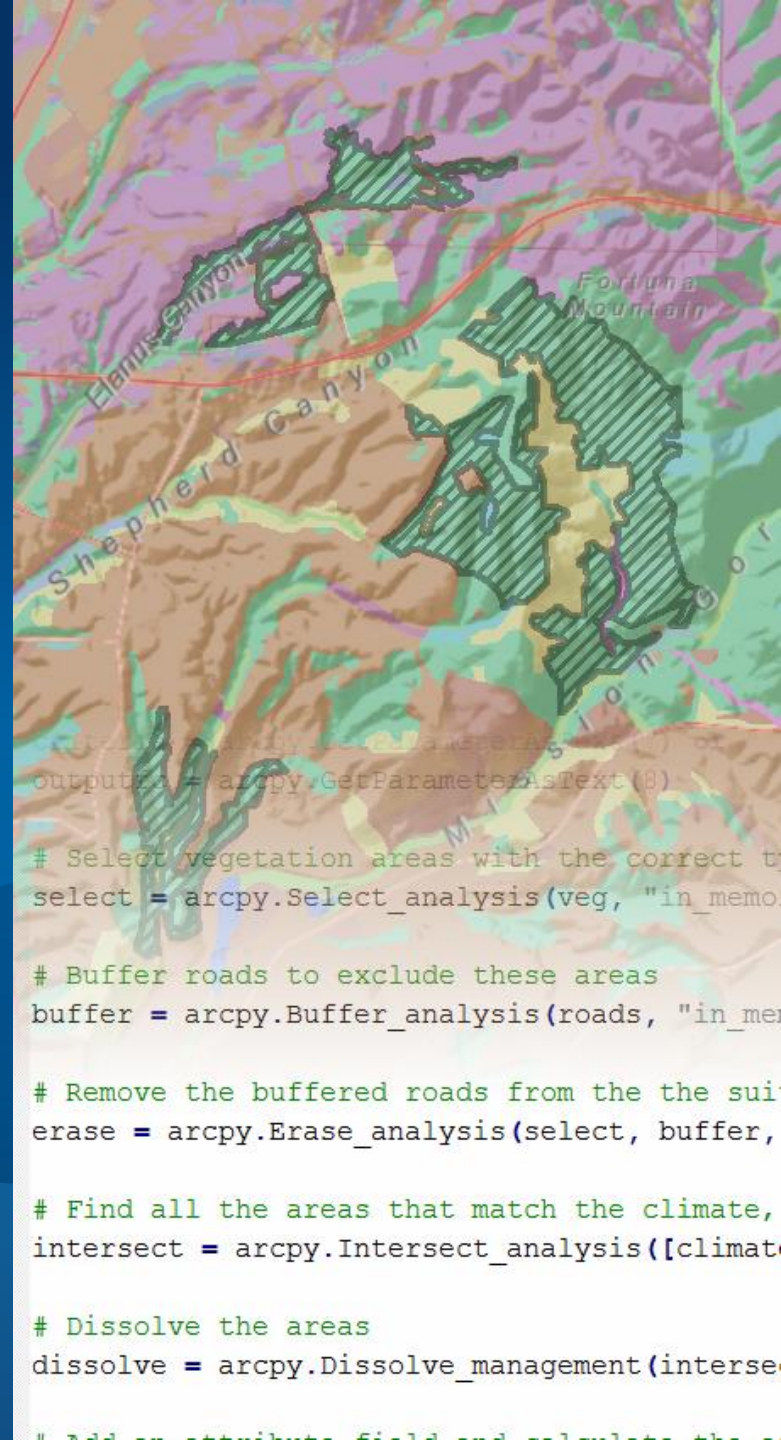


ModelBuilder & Python

- Models can be exported into Python Scripts
- Captures variables in model, often need to modify
- Expand beyond some ModelBuilder limitations (if/else, for loops)



Export Model



Run geoprocessing tools

```
import arcpy
```

Follow tool syntax

```
arcpy.toolname_toolboxalias()
```

Enter input and output parameters

How do I use a specific tool?

Tool help page

Copy as Python Snippet

```
help(arcpy.Buffer_analysis)
```

Syntax

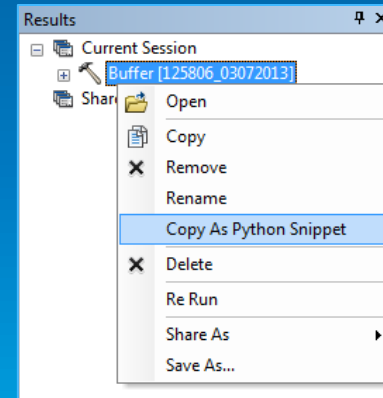
```
Buffer_analysis (in_features, out_feature_class,  
buffer_distance_or_field, {line_side}, {line_end_type},  
{dissolve_option}, {dissolve_field})
```

Code Sample

Buffer Example (Python Window)

The following Python Window script demonstrates how to use the Buffer tool:

```
import arcpy  
arcpy.env.workspace = "C:/data"  
arcpy.Buffer_analysis("roads", "C:/output/majorroadsE
```



Geoprocessing environment settings

Use geoprocessing environments as global parameters

See tool help for honored environments

Productivity and code cleanup

`arcpy.env`

```
arcpy.env.workspace = "C:/Data"  
arcpy.env.extent = "0 0 100 100"  
arcpy.env.outputCoordinateSystem = 4326 #WKID
```


Troubleshooting

Why do errors occur?

Incorrect tool use, typos, syntax,

...or bugs ☹️

My script doesn't work!? Help!

View geoprocessing messages

Use Python error handling

Debug the script in an IDE



Geoprocessing messages

Three types of messages

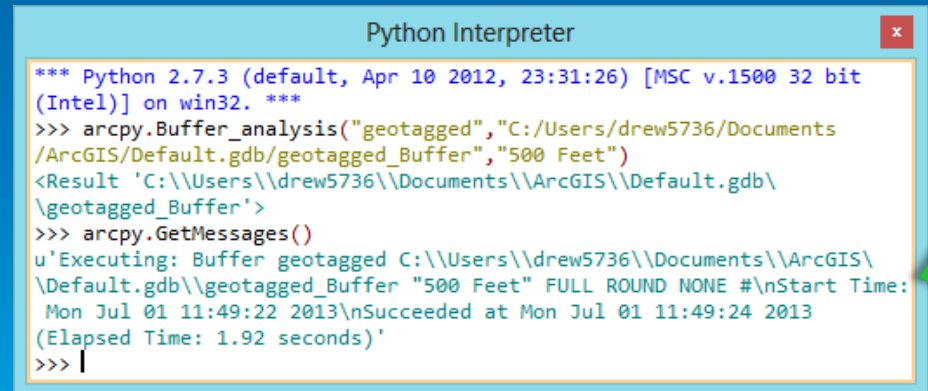
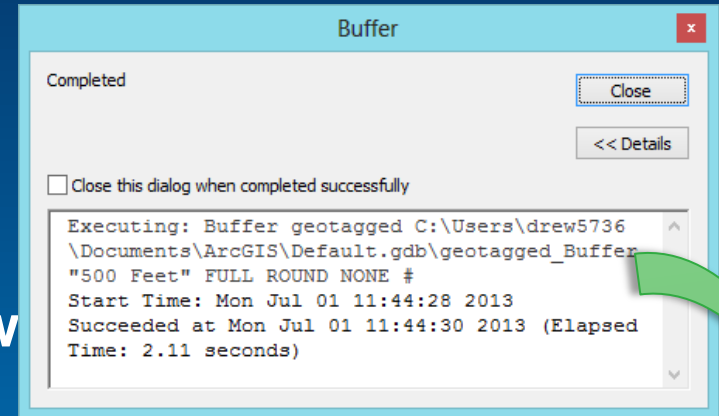
Info, warning, error

Displayed in the Python window

Errors displayed in IDE

To see other messages

`arcpy.GetMessages`



Python error handling

Try...Except...

Try to do something, and if an error occurs, do something else

```
# Start Try block
try:
    arcpy.Buffer_analysis("Roads.shp", ... )
# If an error occurs
except:
    # Print that Buffer failed and why
    print("Buffer failed")
    print(arcpy.GetMessages())
```

Exercise

Writing a Python script

```
23 arcpy.SetProgressorLabel("preparing to
24 time.sleep(5)
25
26 # select vegetation of desired type
27 select = arcpy.Select_analysis(veg, 'in
28     ' "VEG_TYPE" IN (\'Coastal Sage-Cha
29
30 # buffer by 1000 feet
31 road_buff = arcpy.Buffer_analysis(roads
32
33 # remove buffered roads from desired ar
34 erase = arcpy.Erase_analysis(select, ro
35
36 # find interesection of valid climate,
37 intersect = arcpy.Intersect_analysis([c
38
39 # dissolve all adjacent areas together
40 dissolve = arcpy.Dissolve_management(in
41
42 # add field and calculate are of contig
43 arcpy.AddField_management(dissolve, 'Ar
44 arcpy.CalculateField_management(dissolv
45
46 # select out areas that match all crite
47 arcpy.Select_analysis(dissolve, outfc,
48 arcpy.AddMessage('NUMBER OF NON CONTIGU
49
50 arcpy.ResetProgressor()
51 print ('FINISHED')
```

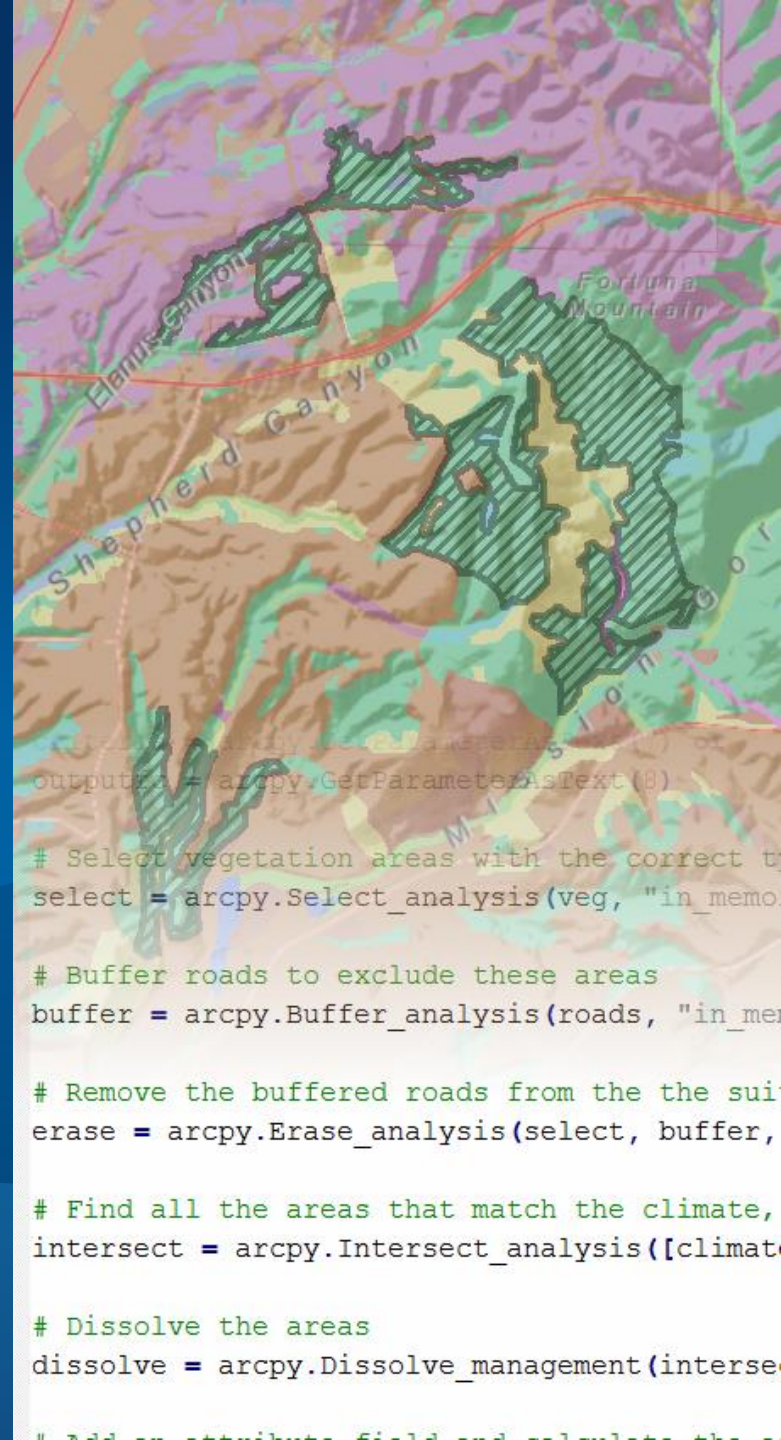
demo4_suitability_analysis_Tool.py x

Python Interpreter

```
*** Remote Python engine is active ***
>>>
*** Remote Interpreter Reinitialized ***
>>>
[Dbg]>>>
```

Call Stack Variables Watches Breakpoints

ArcPy Functions



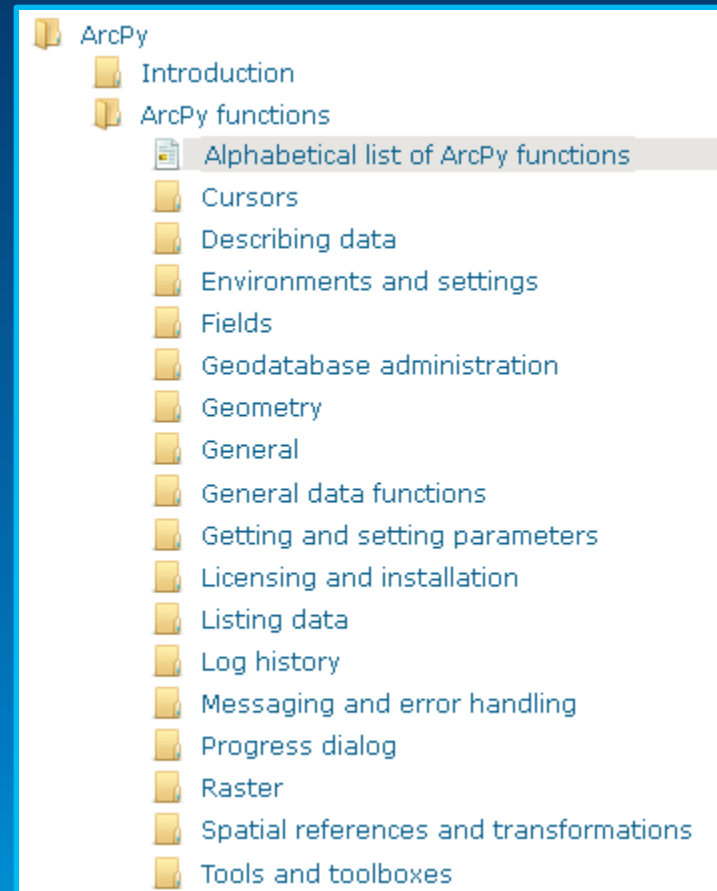
ArcPy functions

Perform useful tasks

List data: **ListFeatureClasses**

Get data properties: **Describe**

Enables **automation** of manual tasks

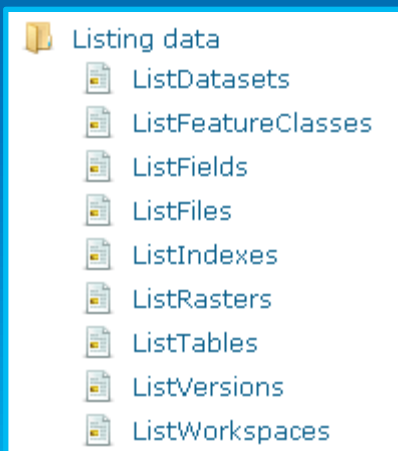


Batch processing

Automating a geoprocessing operation to run multiple times

Clip every feature class in a
to a boundary

Calculate statistics for every
in a folder



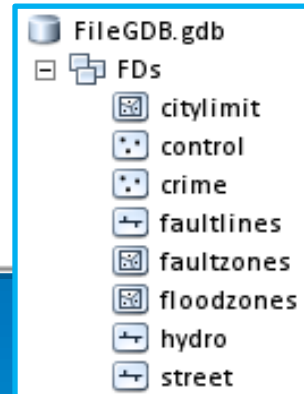
List functions used in Python
to perform batch processing

arcpy.ListFeatureClasses

```
# Set the workspace
arcpy.env.workspace = "C:/Data/FileGDB.gdb/FDs"

# Get a list of all feature classes
fcList = arcpy.ListFeatureClasses()

# Print the names of the feature classes
for fc in fcList:
    print(fc)
```



Getting data properties

Describe function reads data properties

Returns an object with properties

Data type

Shape type

Spatial reference

Fields

```
# Describe a feature class
desc = arcpy.Describe("C:/Data/Roads.shp")

print(desc.shapeType)
>>> "Polyline"
```


Python script tools

Python script as custom
geoprocessing tool

Great way to create and share
workflows, extend ArcGIS

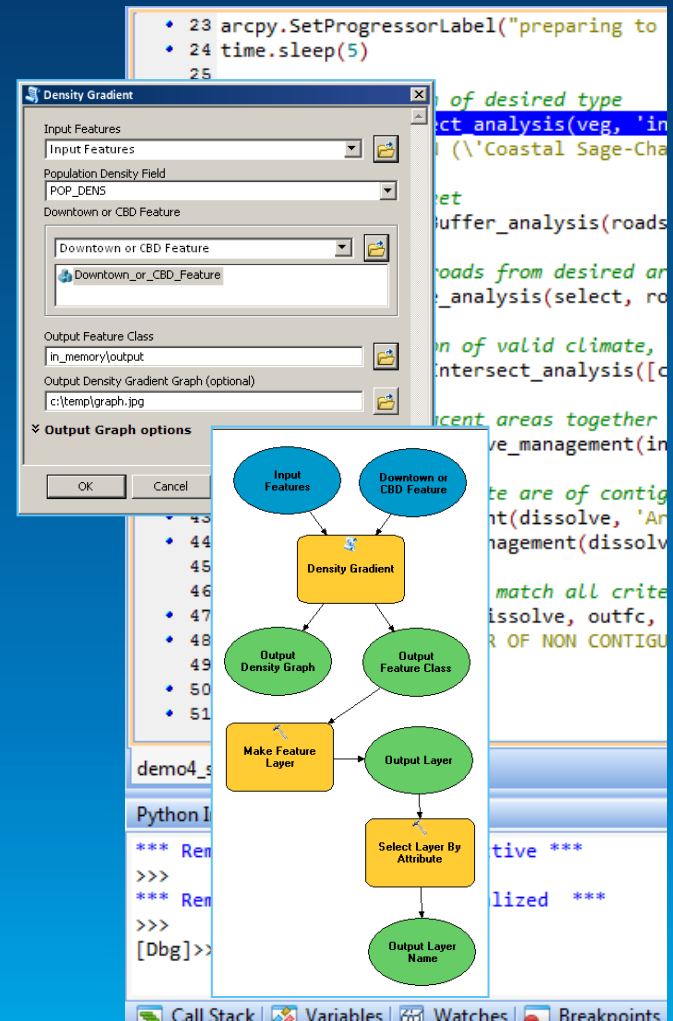
More accessible than stand-alone

Integrated with geoprocessing
framework

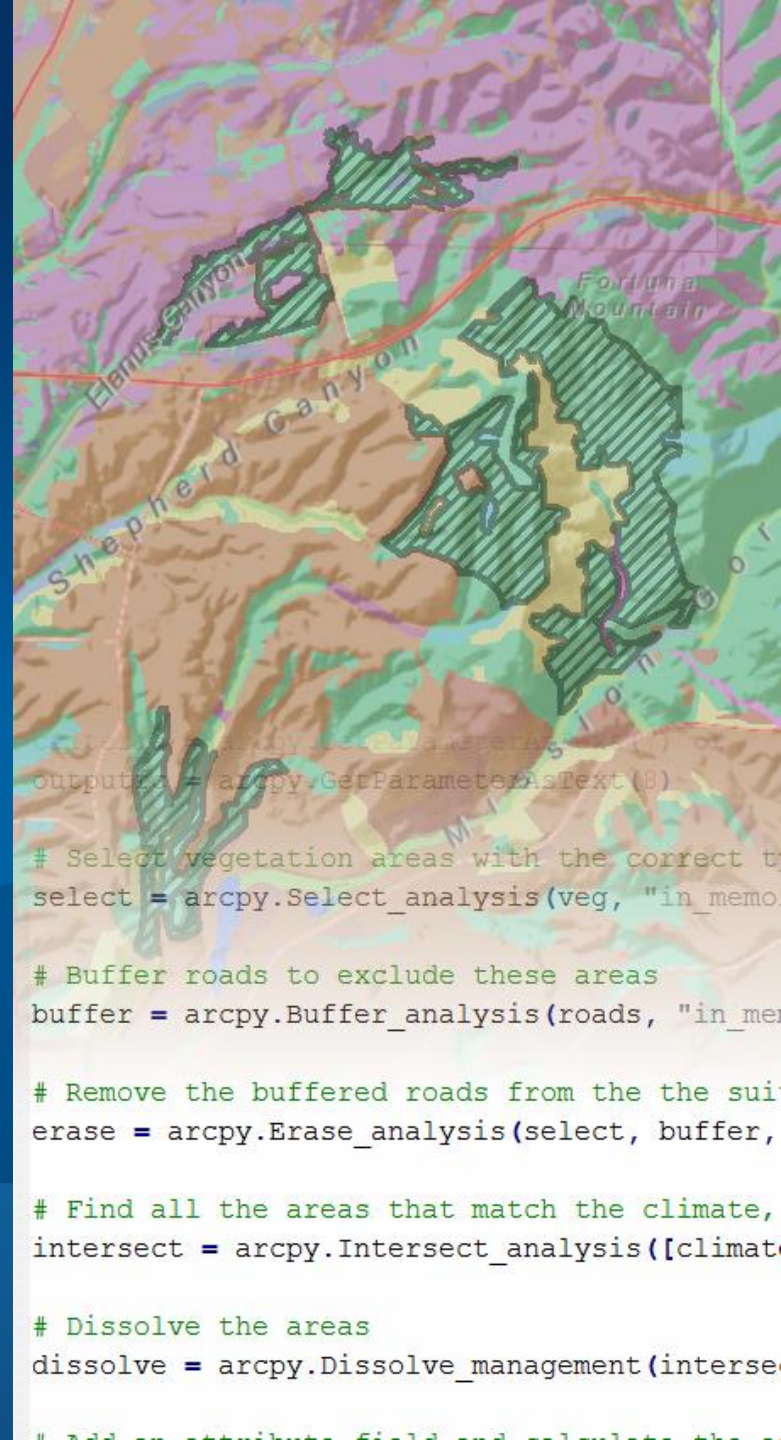
Use it just like any other tool

Geoprocessing properties and
environments

Works with map layers



Create a Script Tool



Resources

[resources.ArcGIS.com](https://resources.arcgis.com)

Analysis, Python

arcpy.wordpress.com

GIS Stack Exchange, Stack Overflow

Python References

[Python Scripting for ArcGIS](#) - Esri Press

[Learning Python](#) – O'Reilly Books

[The Python Standard Library by Example](#) by Hellmann

python.org